웹 환경을 기반으로 한 원격 자원제어 시스템

이 기 준[†] · 정 국 영[††] · 정 채 영[†††]

요 약

인터넷의 발전과 더불어 이를 이용한 TCP/IP기반의 원격 자원 시스템은 네트워크에 연결된 모든 형태의 자원들에 대하여 시간과 공간의 제한없이 접근하여 제어할 수 있는 방안을 제시하여 준다. 본 논문에서는 웹 원격 제어 시스템을 구성하기 위하여 네트워크 기반의 서비스 이용자, 서비스 제공자, 서비스 관리자로 구성된 시스템 연합체를 구성하고 이들 삼자간의 상호통신을 통한 자원 제어방식에 대하여 연구하고자 한다. 이때 서비스의 사용자는 자원을 요구하는 사용자이며, 서비스 제공자는 네트워크상에서 서비스를 제공하여줄 수 있는 각종 기기, 디지털 전자제품이 된다. 또한 서비스 관리자는 시스템 전체를 관리하고 운용하며 서비스 요구자와 제공자를 상호연결시켜주는 역할을 수행한다. 구현된 시스템은 실험을 통하여 제안방식의 유연성과 안정성 및 확장가능성에 대하여 확인한다.

Remote Resource Control System based on Web Environment

Kee-Jun Lee[†] · Guk-Yeoung Jung[††] · Chai-Yeoung Jung[†††]

ABSTRACT

TCP/IP based remote resource system using internet presented the method to access and control all the types of resources connected to a network system over time and space. In this paper, We studied a resource control method through a mutual communication of a system association composed of service user, provider, and manager of the network system for the web based remote control system. The service users require the resources, and its providers are a variety of devices and digital electronic products which can provide the services in a network system. The service manager who operates the whole system performs the role to interconnect its users with its providers. Through an experiment, an implemented system confirmed flexibility, stability, and extensibility of the method presented in this paper.

키워드 : TCP/IP, 원격 자원제어, 시스템 연합체, jini Technology

1. 서 론

최근 하드웨어 기술의 성장과 더불어 인터넷의 급속한 발전은 기존의 인간생활에서 영위할 수 없었던 시간과 공간을 초월한 새로운 생활문화를 만들어 가고 있다. 초기의 인터넷 사용자들은 단순한 정보의 수집과 배포를 목적으로 사용하였지만 이제는 네트워크상에 연결된 수많은 컴퓨터 자원을 활용하며, 유·무선방식을 이용하여 원거리에 떨어져 있는 각종 기기나 디바이스, 가전 네트워크 시스템, 디지털 전자제품들을 제어하며 이용할 수 있는 기술에 대한 요구들이 날로 높아져가고 있다. 이런 다양한 요구들을 충족시키기 위해서 분산 네트워크상의 이질적 시스템간의 상호작용을 지원해주는 연구와 함께 이를 접목한 다양한 서비스개발에 관한 연구들이 활발히 진행되고 있다[1, 2].

특히 유·무선통신을 이용한 기기들간의 상호교류 및 접근, 제어에 관한 연구들이 진행되고 있는데 이와 관련된 근거리 무선 접속기술[3]로 IrDA(Infrared Data Association), 무선 LAN(IEEE802.11), SWAP(Shared Wireless Access Protocol), 블루투스(Bluetooth)[4]등이 있으며 이중 블루투스는 근거리 무선통신의 세계적인 표준으로 인정을 받고 있다. 블루투스는 2.4㎓대 ISM(Industrial, Scientific, Medical) 대역 주파수를 사용하며 1Mbps의 속도로 최대 10m내에서 각종 단말기들을 무선 접속해 사용할 수 있는 기술로써 짧고 유연성이 좋은 데이터 패킷을 사용하기 때문에 접속률을 극대화시킬 수 있는 장점을 지니고 있다. 또한 유선 접속기술로는 MS의 UPNP[5]와 Sun의 Jini Technology[6-9]이 있다. UPNP는 PNP(Plug & Play)기술을 네트워크상으로 확장한 것으로 기기들간의 복잡한 환경설정 작업을 생략하고 각종 디바이스를 네트워크에 접속만하면 IPP(Internet Printing Protocol)네트워크 프로토콜을 이용하여 자동적으로 디바이스를 찾아 사

※ 본 논문은 2000년도 조선대학교 학술연구비의 지원을 받아 연구되었음.
† 정 회 원 : 조선대학교 대학원 전산통계학과
†† 준 회 원 : 동광대학교 컴퓨터정보과
††† 종신회원 : 조선대학교 자연과학대학 전산통계학과
논문접수 : 2001년 2월 26일, 심사완료 : 2001년 6월 18일

용할 수 있는 기술이다. 그리고 Jini Technology는 썬 마이크로시스템즈에서 제안하고 있는 접속 기술로서 자바를 기반으로 다양한 방식(LAN, ADSL, MODEM, 전력선, 무선 등)으로 네트워크에 접속되어 있는 지능형 기기(마이크로 프로세서가 장착되고 지니 아키텍처가 적용된 기기)들이나 소프트웨어들이 동적으로 상호 작용을 할 수 있는 기술이다.

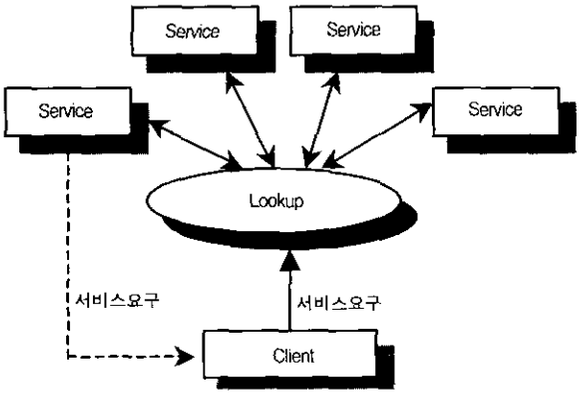
본 논문에서는 Jini Technology를 기반기술로 하여 웹 상에서 원격으로 자원을 제어할 수 있는 원격 자원 제어 시스템에 관한 연구를 수행하고자 한다. 원격 자원 제어 시스템은 네트워크상에 연결된 여러 형태의 기기나 디바이스에 대하여 시간과 공간의 제한없이 접근, 제어할 수 있는 방안을 제시하여 준다. 원격 자원 제어 시스템은 자원을 요구하는 서비스 이용자(Client), 자원을 제공하여주는 서비스 제공자(Service), 시스템 전체를 관리하고 운용하며 서비스 이용자와 제공자를 상호연결시켜주는 역할을 수행하는 서비스 관리자(Lookup)로 구성된 시스템 연합체를 구성하며 이 연합체내의 상호 교류와 접근을 통하여 상호 자원을 제어 및 이용할 수 있는 메카니즘을 제공하여 준다.

본 논문은 다음과 같이 구성되어 있다. 먼저 제2장에서는 웹 원격 자원 제어 시스템의 구조 및 구성요소에 대하여 살펴보고, 제3장에서는 구성된 시스템 연합체내의 상호 교류와 통신방식에 대하여 기술한다. 그리고 제4장에서는 제어 시스템을 구현하고 실험을 통하여 제안된 웹 원격 자원 시스템의 유용성과 안정성, 확장가능성을 확인하며 마지막으로 제5장에서 본 논문에 대한 결론을 맺는다.

2. 웹 원격자원 제어 시스템

2.1 제어 시스템

구성된 제어 시스템은 크게 세 가지의 서브 어플리케이션으로 구분하여 서비스 사용자(Client), 서비스 제공자(Service), 서비스 관리자(Lookup)로 구성된 시스템 연합체를 구성한다. 이때 서비스의 사용자는 자원을 요구하는 주체이며, 서비스 제공자는 네트워크상에서 자원을 제공하여줄 수 있는 각종 기기, 디지털 전자제품이 된다. 또한 서비스 관리자는 시스템 연합체를 관리하고 운용하며 서비스 사용자와 제공자를 상호 연결시켜주는 작업을 수행한다. (그림 1)은 구성한 웹 자원 제어 시스템 연합체의 구성도이다.



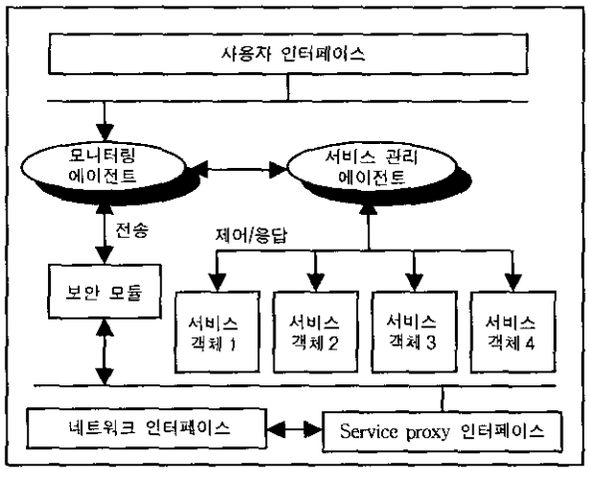
(그림 1) 웹 자원 제어 시스템 연합체 구조

제어 시스템 연합체를 구성하고 있는 각 Service들은 자신의 서비스 객체를 Lookup에 등록을 하고, 이를 이용하는 Client는 Lookup에 등록된 Service의 서비스 객체 중 원하는 자원을 제공하여주는 Service 목록을 검색한 후 공유된 자원을 이용할 수 있다.

2.2 제어 시스템의 구성

2.2.1 Service

Service는 Client로부터 요구되어진 자원의 요구를 실행할 수 있도록 자신의 자원을 제공하여준다. (그림 2)는 Service의 구성모듈로 시스템에 존재하는 에이전트간의 관계를 나타내고 있다.



(그림 2) Service 모듈의 구조

Service를 구성하고 있는 각 에이전트 및 컴포넌트의 기능은 다음과 같다.

- **보안 모듈**

개방된 웹상에 존재하는 웹 자원 제어시스템은 요구된 제어패킷의 유효성여부를 확인하기 위하여 보안모듈을 통해 인증을 받게된다. Client로부터 전송된 제어패킷은 암호 해시를 통하여 축약된 메시지로 변환되며 보안모듈에서는 축약메세지의 데이터와 비교하여 제어 패킷의 유효성여부를 확인하게 된다. 인증된 제어패킷은 모니터링 에이전트에게 전송되어 Service의 자원을 이용하게 된다.

- **모니터링 에이전트**

모니터링 에이전트는 Service를 구성하고 있는 각 에이전트에 대한 정보를 관리하는 에이전트로 각 에이전트 이

름과 주소, 에이전트의 흥미와 노력에 따라 통신하는 에이전트를 지정한다. 또한 보안 모듈을 통하여 전달된 Client의 제어패킷을 해석하여 서비스관리 에이전트에 전달하고 수행결과를 Client에게 전달하는 역할을 수행한다.

- **서비스 관리 에이전트**

서비스관리 에이전트는 모니터링 에이전트와 서비스 객체에 대한 중개 창구가 되는 에이전트이다. 중개 에이전트는 모니터링 에이전트로부터 제어 패킷의 내용에 대하여 서비스 객체의 대행조회를 요청하고, 만일 직접적인 자원의 제어가 필요할 경우 서비스 객체에게 작업을 요청한다.

- **서비스 객체**

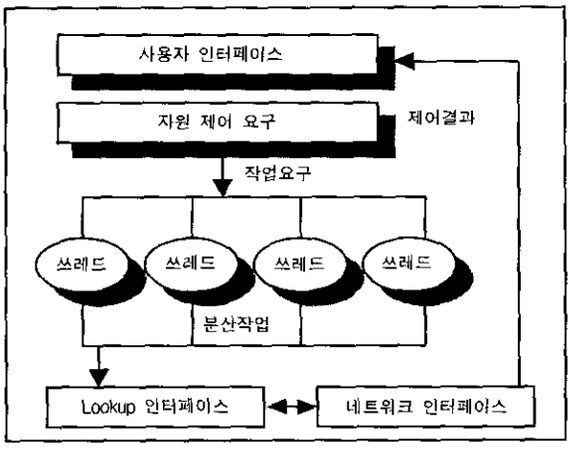
Client로부터 서비스 요청이 있을 때 실행되는 프로그램 코드블록이다. Service 사용자 인터페이스를 통한 서비스 허용 요청이 있을 때 Service Proxy 인터페이스를 통하여 Lookup에 등록된다

- **Service Proxy 인터페이스**

Service Proxy 인터페이스는 Client로부터 자원제어작업요구를 수행할 수 있도록 Lookup 에 서비스 객체를 등록하는 모듈이다. 이때 서비스 객체는 작업 수행을 위해 Service Proxy 인터페이스를 통하여 Lookup에 등록된다. 서비스 객체를 등록하려면 Service Proxy 인터페이스는 웹 상에 존재하는 Lookup의 위치를 파악하여야 하는데 만일 서버의 위치(IP)를 알고 있다면 유니캐스트(Unicast) 방식을 이용하여 Proxy를 등록하고 서버의 위치를 알지 못한다면 멀티캐스트(Multicast) 방식을 이용하여 Lookup의 위치를 파악하여 Proxy를 등록한다.

2.2.2 Client

Client는 구축된 시스템 연합체의 각 Service들에게 자원제어요구를 수행하는 주체로 모든 Service들에게 분산작업을 요청한다. (그림 3)은 Client의 모듈의 구조를 나타낸다.



(그림 3) Client 모듈

Client는 사용자 인터페이스, 자원제어요구 모듈, 네트워크 인터페이스, Lookup 인터페이스로 구성되며 각 모듈의 수행방식은 다음과 같다.

- **사용자 인터페이스**

사용자 인터페이스는 사용자의 제어요구를 입력받아 자원제어 모듈에 전달한다. 또한 Service로부터 전달받은 제어결과를 사용자에게 전달하는 모듈이다. 사용자 인터페이스 환경은 어플리케이션 환경 또는 웹 브라우저 환경으로 구성된다.

- **자원제어요구 모듈**

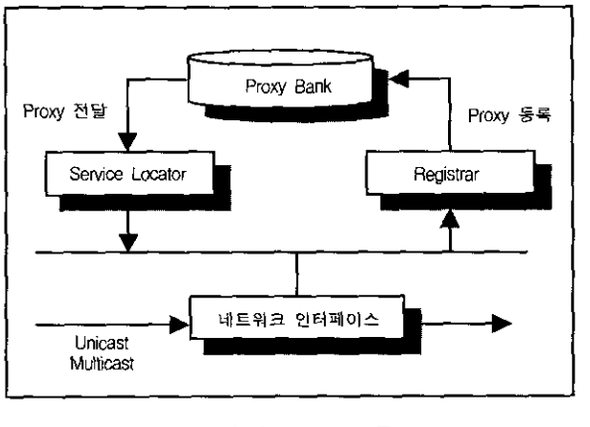
사용자 인터페이스로부터 전달받은 제어요구를 Service에게 요구하는 모듈로 먼저 lookup 인터페이스를 통하여 확보된 Service의 목록에서 제어대상이 되는 Service에 비례하는 쓰레드를 발생시켜 분산 작업을 시도한다. 각기 실행된 쓰레드는 제어패킷에 대한 메시지 축약과정을 거쳐 해당 Service에게 전달하고, Service로부터 제어 결과를 전달받아 사용자 인터페이스에 전달한다.

- **Lookup 인터페이스**

Lookup 인터페이스는 Lookup에 등록된 Service의 Proxy 목록을 전달받는 역할을 수행한다. 전달받은 목록 중 입력된 자원제어 대상 Service의 목록을 추출한 후 자원제어요구 모듈에 전달한다. 자원제어요구 모듈은 전달받은 목록에 따라 쓰레드를 발생하고 분산작업을 시도한다. Lookup 인터페이스는 Service의 목록을 수집하기 위하여 Looup의 위치(IP)를 찾는다. 만일 Looup의 위치를 파악하고 있으면 유니캐스트(Unicast) 방식을 이용하고, Lookup의 위치를 모르는 경우 멀티캐스트(Multicast) 방식을 이용하여 위치를 확보한다.

2.2.3 Lookup

Lookup은 네트워크 상에 연결되어 있는 독립된 호스트에 존재하며 Client에게 Service를 중개하는 역할을 수행한다.



(그림 4) Lookup 모듈

그리고 Service들은 Lookup에게 자신의 서비스객체와 자신을 설명하는 속성들을 등록한다. 또한 Client는 Lookup에서 원하는 서비스객체와 속성을 찾고 이를 다운로드하여 사용한다. (그림 4)는 Lookup의 모듈을 나타낸다.

Lookup은 Proxy Bank, Service Locator, Registrar, 네트워크 인터페이스 모듈로 구성되어 있고 각 모듈의 기능은 다음과 같다.

- **Proxy Bank**

Service로부터 전달받은 서비스객체를 저장하는 창고역할을 수행한다. 저장된 서비스 객체들은 Client의 요청에 의해 해당 Client에게 전달되고 Client와 Service를 중개시킨다.

- ***Service Locator***

Service Locator는 Client로부터 요청받은 서비스의 객체가 Proxy Bank 모듈에 저장되어 있는가를 검색하고 만일 객체가 존재하고 있을 경우 Proxy 객체를 Client에게 전송한다.

- **Registrar**

Registrar 모듈은 서비스를 수행할 Service로부터 객체(proxy)를 전달받아 Proxy Bank에 저장하는데 전달받은 Proxy 객체는 Service에서 수행할 수 있는 사용자 정의 프로그램 모듈로 Client에게 전달되어 서비스를 수행한다.

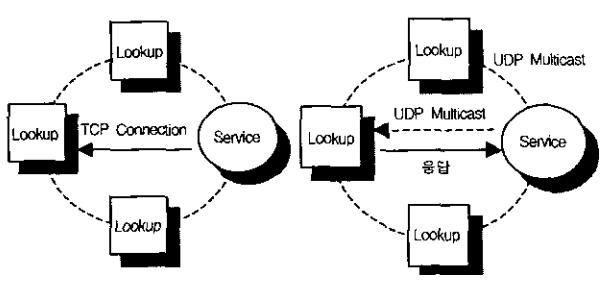
- **네트워크 인터페이스**

네트워크 인터페이스는 Unicast 방식과 Multicast 방식으로 수행되는 Client와 Service의 Proxy 인터페이스에게 현재 Proxy Bank에 저장된 Proxy 객체의 모듈 목록을 전달하고 객체(proxy)를 전달받는다.

3. 제어 시스템 연합체의 동작 방식

3.1 Service의 등록

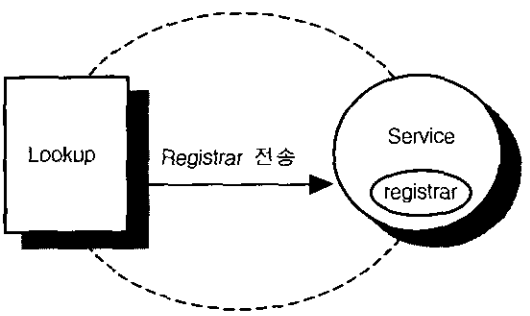
시스템 연합체를 구성하고 있는 각 Service는 자원제어 서비스하기 위해서 자신의 서비스 객체(Proxy)를 등록시켜야 한다. 이때 Service가 서비스객체를 Lookup에 등록하는 과정은 다음과 같다.



(그림 5) Service의 Lookup위치 파악

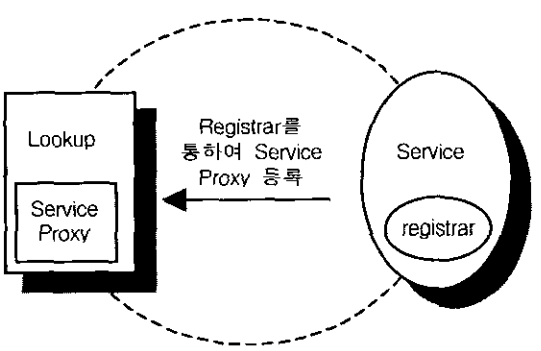
【과정 1】 먼저 서비스 객체(Proxy)를 등록시킬 Lookup을 검색하는데 이 때의 과정을 Discovery라 한다. 만일 Lookup의 위치를 알고 있다면 Unicast를 사용하여 TCP 연결을 실시하고 Lookup Server의 위치를 알지 못하면 UDP Multicast request를 이용하여 Lookup의 위치를 검색한다. 이 경우 네트워크 상에는 다수의 Lookup이 존재할 수 있는데, Service가 Lookup을 찾으려고 UDP Multicast request를 하게 되면 Lookup Server는 이에 대한 응답을 하게 된다.

【과정 2】 Service는 위치가 파악된 Lookup으로부터 Registrar를 얻어 온다. Service가 Unicast를 이용하여 직접적으로 Lookup에 TCP 연결을 하거나 또는 UDP Multicast request를 이용하여 요청신호를 보내면 Lookup은 Listening을 하고 있다가 해당 Service에게 registrar를 전송한다. 이때 registrar은 Service가 Lookup에 접근하기 위한 접근 Proxy 역할을 수행한다.



(그림 6) service에 registrar 전송

【과정 3】 Service는 서비스 객체(Proxy)를 전송받은 registrar를 통하여 Lookup에 전송하는데, 이 전송과정을 통하여 Service는 자신의 서비스 객체를 Lookup에 등록한다.

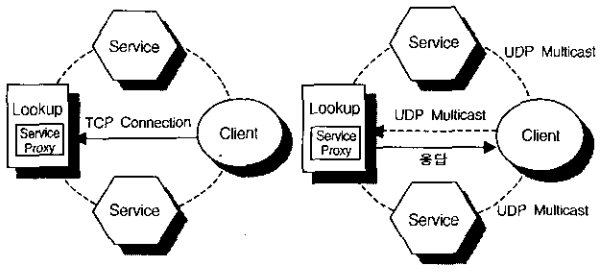


(그림 7) 서비스 Proxy 등록

3.2 Client의 서비스 요구

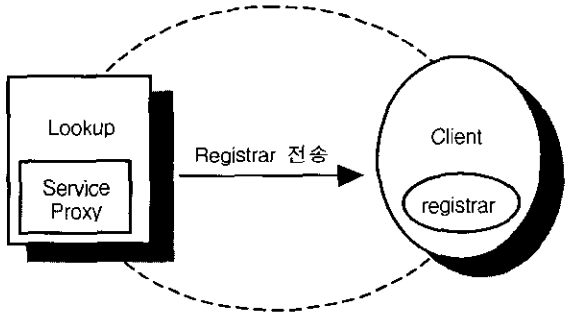
Lookup에 대한 Service들의 서비스 등록작업을 통하여 자원들은 Client의 서비스 요구에 의하여 제어되는데, Client들은 Lookup에 등록된 Service의 객체를 전달받아 이용한다. Client의 서비스 요구과정은 다음과 같다.

【과정 1】 Client도 Service와 마찬가지로 동일한 매커니즘을 이용한다. 서비스 객체를 등록시킬 Lookup의 위치를 파악하는데 이 과정을 Discovery라 한다. 만일 Lookup의 위치를 알고 있다면 Unicast를 사용하여 TCP 연결을 실시하고 Lookup Server의 위치를 알지 못하면 UDP Multicast request를 이용하여 Lookup의 위치를 검색한다.



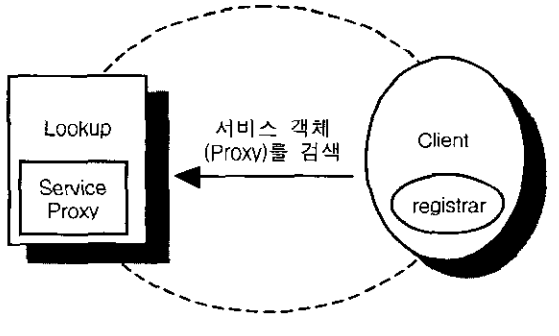
(그림 8) Client의 Lookup 위치 파악

【과정 2】 Lookup에서는 서비스를 요청하는 Client가 있을 경우 나중에 Client가 자신에게 접근할 수 있는 접근 객체로 Registrar를 전송하여 준다.



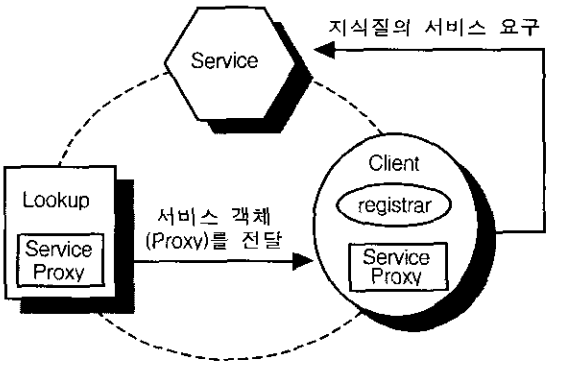
(그림 9) Client에 registrar 전송

【과정 3】 Client는 전송 받은 Registrar를 이용하여 원하는 서비스 객체(Proxy)가 Lookup에 등록되어 있는지를 검색하는데 이 과정을 Lookup이라 한다.



(그림 10) 서비스 객체(Proxy) 검색

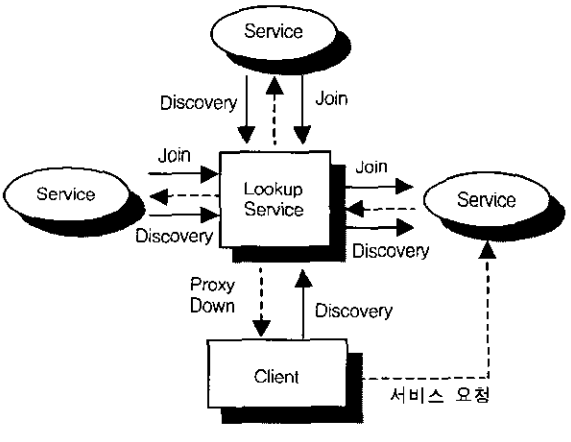
【과정 4】 Lookup는 Client가 원하는 서비스 객체(Proxy)를 전송하여 준다. Client는 전송받은 서비스 객체를 자신의 자원제어요구 모듈에서 실행하여 해당 Service에게 제어패킷을 전달한다.



(그림 11) 객체전달 및 자원제어서비스 요구

3.3 기반 기술 - Jini Technology[6-9]

지니는 SUN사에서 제안하는 네트워크 PnP(Plug & Play) 기술로 네트워크에 접속된 기기나 소프트웨어들을 동적으로 상호작용 가능케 하는 런타임 인프라스트럭처 기술로 지니는 이용자(Client)와 제공자(Service), 관리자(Lookup)로 구성되며 각각의 구성요소들은 Discovery, Join, 그리고 Lookup 프로토콜을 이용하여 상호 통신하고 이용자와 제공자는 기본적으로 자바 RMI(Remote Method Invocation)를 이용하여 상호작용 한다. (그림 12)는 지니 기술의 수행과정을 나타낸다.



(그림 12) Jini Technology

지니 시스템의 수행과정은 먼저 서비스 제공자가 Discovery 프로토콜을 이용하여 서비스 관리자를 찾고, Join 프로토콜을 이용하여 서비스 관리자에게 서비스와 상호작용을 하는 직렬화된 자바 객체인 프록시 객체를 등록한다. 그러면 서비스 이용자는 Discovery 프로토콜을 이용하여 서비스 관리자를 찾고 Lookup 프로토콜을 이용하여 자신이 원하는 서비스 목록을 검색하여 프록시 객체를 다운로드 받은 다음 서비스와 상호작용을 한다.

4. 실험 및 고찰

제안한 시스템은 웹 환경에서 자원의 제어와 공유를 수행할 수 있는 시스템이다. 구현 및 실험을 통하여 확장가능성 및 유용성에 대하여 고찰하고자 한다.

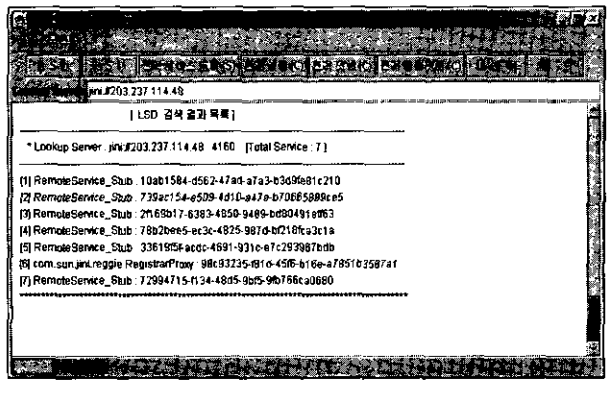
4.1 구현환경

시스템 연합체에는 Client 1대, Lookup 1대, Service 3대를 이용하여 실험하였다. 시스템의 사용자 인터페이스, 자원제어 모듈은 JDK 1.3으로 구현하였고, 시스템 연합체 구축기술로 *Jini Starter Kit Ver 1.1 Beta 2*를 이용하였다. 시스템 연합체를 구성하는 각 자원들은 모두 고유 IP를 가지고 인터넷에 연결되어 있으며, 실험의 내용은 자원 제어 시스템의 구축방안과 구축된 환경에서의 자원 제어의 유용성에 대하여 실험하였다.

4.2 자원 제어 시스템의 구축

4.2.1 서비스의 서비스 객체 등록

시스템 연합체를 구축하기 위해서는 먼저 Lookup이 가동되어야 한다. Lookup은 웹 서버(Web Server), Rmid, Lookup Server에 의해 구동되어지며 (그림 13)는 Lookup이 구동된 후 자원제어 서비스를 제공해주는 Service들이 등록된 모습이다.



(그림 13) Service 객체의 등록

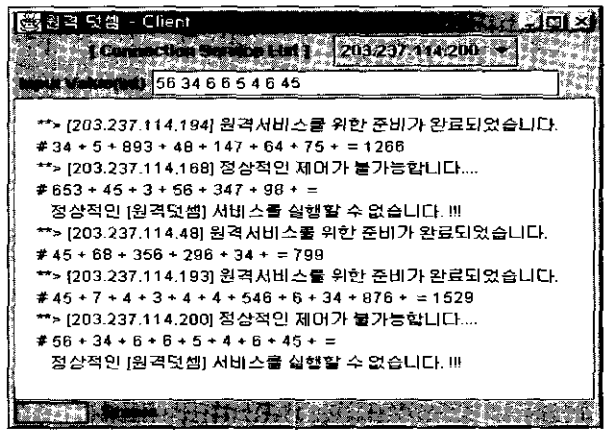
구동된 Service는 서비스 객체를 등록할 Lookup의 위치를 파악하기 위하여 Discovery 작업을 수행한다. 위의 (그림 13)에서 파악된 Lookup의 위치(IP)는 "203.237.114.48"이며 현재 등록된 서비스는 총 7개로 이중 RegistrarProxy 서비스는 Service나 Client로 전송되어 서비스 객체의 전송을 지원하는 Lookup의 기본 서비스 객체이다.

RegistrarProxy를 제외한 RemoteService_Stub 서비스는 자원 원격 제어를 위하여 각 Service로부터 등록된 서비스 객체들이며, 서비스 명칭뒤의 코드들은 Lookup에서 부여된 서비스 코드이다.

4.2.2 Client의 서비스 요구

구성된 시스템 연합체는 우선적으로 안정성을 유지하여야 한다. 안정성을 확보한다함은 Client의 제어서비스 작업중 플랫폼의 고장이나 네트워크의 단절등 예기치 못한 사건들에 대하여 유연함을 지녀야한다. 따라서 제어 서비스를 실행하고 있는 Client는 Servce의 상태여부를 점검하고 이에 적절한 서비스를 제공받아야 한다.

*Client*의 간단한 덧셈 요구작업을 통하여 제어 시스템의 안정성 및 서비스 요구 메카니즘을 살펴본다. (그림 14)는 Client에 의해 요구되어지는 간단한 덧셈의 작업결과이다.



(그림 14) 간단한 덧셈의 작업요구

**> [203.237.114.194] 원격서비스를 위한 준비가 완료되었습니다.
34 + 5 + 893 + 48 147 + 64 + 75 + = 1266
**> [203.237.114.168] 정상적인 제어가 불가능합니다.
653 + 45 + 3 + 56 + 347 + 98 + =
정상적인 [원격덧셈] 서비스를 실행할 수 없습니다. !!!
**> [203.237.114.48] 원격서비스를 위한 준비가 완료되었습니다.
45 + 68 + 356 + 296 + 34 + = 799
**> [203.237.114.193] 원격서비스를 위한 준비가 완료되었습니다.
45 + 7 + 4 + 3 + 4 + 4 + 546 + 6 + 34 + 876 + = 1529
**> [203.237.114.200] 정상적인 제어가 불가능합니다....
56 + 34 + 6 + 6 + 5 + 4 + 6 + 45 + =
정상적인 [원격덧셈] 서비스를 실행할 수 없습니다. !!!

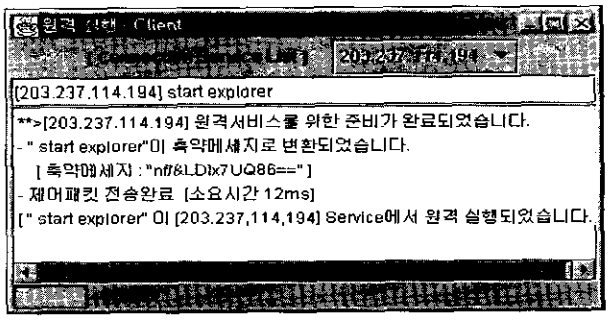
덧셈 Client에서 입력된 수치들은 선택된 Service에게 덧셈작업을 요구한다. 이때 Client는 위치가 파악된 Lookup으로부터 해당 Service의 서비스 객체를 다운로드 받아 이 서비스 객체를 이용하여 Service들에게 작업을 요구한다. 위의 (그림 14)에서는 "203.237.114.48", "203.237.114.193", "203.237.114.194"서비스들은 원격 덧셈 서비스를 수행하였지만 "203.237.114.168", "203.237.114.200" 서비스는 해당 시스템이나 네트워크 상의 문제로 인하여 원활한 서비스를 제공할 수 없는 상태를 나타내고 있다. 기존의 2-tier 에서는 Client가 Server에서 작업을 요구하였을 때 Server에 문제가 발생하면 Client는 작업이 중지되거나 심각한 경우 재가동하여야 하는 문제점이 발생하였지만 제어 시스템에서는 서비스의 존재유무, 작동여부에 관여하지 않고 Lookup의 중재역할로 작업을 수행할 수 있는 다른 Service에게 작업을 연동할 수 있다.

4.3 시뮬레이션

4.3.1 원격 프로세스 제어

Client는 Lookup에 등록된 서비스 객체를 이용하여 시스템 연합체를 구성하고 있는 각 Service의 프로세스를 원격으로 실행하고 제어할 수 있다. Service의 보안모듈은 Client로부터 전달된 제어패킷의 유효성을 검증하기 위하여 전달된 제어패킷을 축약메세지의 데이터와 비교하여 유효성여부를 확인한다. 인증된 제어패킷은 해당 에이전트에 전달되어 Service에서 실행되고 제어되어진다.

(그림 15)는 Client가 선택된 Service에게 프로세서의 제어를 위한 원격제어 모습이다.



(그림 15) Client의 원격제어 요구

```
**>[203.237.114.194] 원격서비스를 위한 준비가 완료되었습니다.
- " start explorer"이 축약메세지로 변환되었습니다.
  [ 축약메세지 : "nf/&LDlx7UQ86==" ]
- 제어패킷 전송완료 [소요시간 12ms]
[" start explorer" 이 [203.237,114,194] Service에서 원격 실행되었습니다. ]
```

Client는 "203.237.114.194" Service에게 "start explore"이라는 제어문을 축약메시지로 변환하여 전송한다. 이때 축약메시지로 변환하고 이를 전송하기 위하여 소요된 시간은 12ms이다. 전송된 메시지는 Service에서 인증과정과 해석과정을 통하여 해당 프로세스를 생성한다.

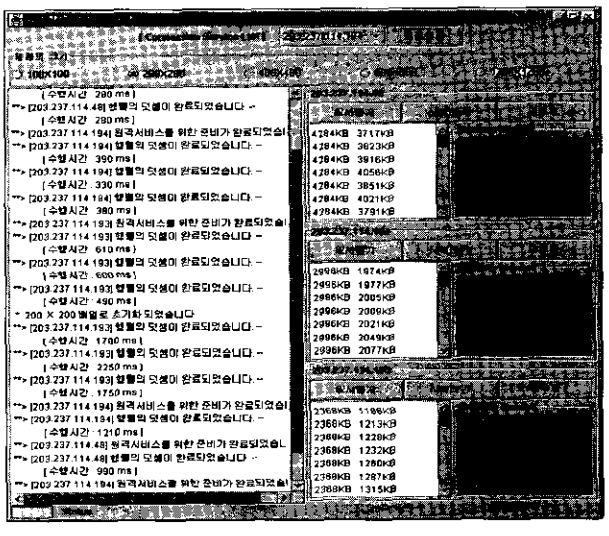
4.3.2 원격 행렬 연산

원격 행렬 연산 실험은 시스템 연합체를 구성하는 Service의 자원을 이용하여 Client가 요구하는 행렬의 연산을 수행하는 실험이다. 이를 통하여 구성된 시스템 연합체가 원거리의 자원을 효율적으로 이용할 수 있는 가능성을 제시하고자 한다. 실험에 참여하는 Service들은 총 3대를 사용하였고 각 시스템의 사양은 다음과 같다.

〈표 1〉 실험에 사용된 Service의 사양

Client	Service 1	Service 2	Service 3
pII 433 96MB	pII 433 96MB	pIII 600 128MB	p200 64MB
WinME	WinME	WinME	Win98

실험을 위하여 작업을 요구하는 Client에서는 100×100, 200×200, 400×400, 800×800, 1200×1200 크기의 배열을 작성하여 Service에게 작업을 요청한다. 배열연산 요청을 받은 Service는 작업크기에 해당하는 메모리를 확보하고 이를 연산하여 Client에게 전송한다. (그림 16)은 원격배열 작업을 수행하고 있는 Client의 모습이다.

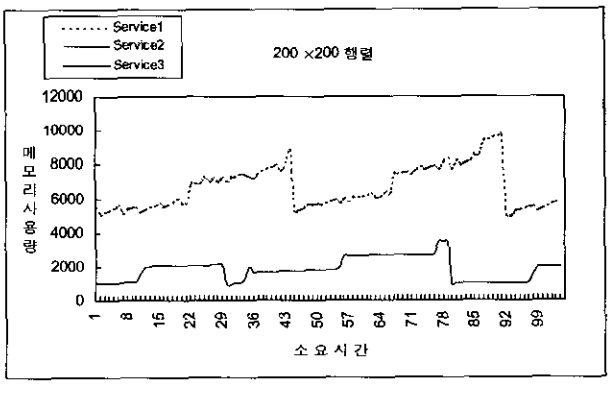


(그림 16) 원격 행렬 연산

각 Service에 대하여 행렬의 연산을 요구하여 다음과 같은 결과를 얻었다. <표 2>는 작업요구를 받은 Service들의 수행시간을 나타내며 (그림 17)은 각 행렬의 크기에 의한 Service들의 메모리 사용량을 나타낸다.

〈표 2〉 Service들의 수행시간

행렬의 크기 \ Service	Service 1	Service 2	Service 3
100×100	272ms	372ms	606ms
200×200	912ms	1476ms	1944ms
400×400	3218ms	5178ms	7604ms
800×800	14443ms	19790ms	29240ms
1200×1200	96220ms	107650ms	181610ms



(그림 17) 200×200 행렬의 메모리 사용량

Client는 Service1이 구동된 플랫폼상에서 실행하여 각각 Service1, Service2, Service3에게 작업을 요구한다. 따라서 Service1의 작업 수행시간은 로컬 네크워크상에서 작업을 요구한 시간과 동일한 결과를 얻었다. Service1의 작업소요시간을 기준으로 보면 Service2의 시스템 사양은 Service1 보다 우수하지만 네트워크를 통한 데이터의 이동시간 때문에 전체적인 작업시간이 늦게 완료되었고Service3 역시 Service1에 비하여 늦은 계산속도를 나타내었다.

(그림 17)의 그래프는 200×200 행렬 연산시 각 Service의 메모리 사용량을 나타내고 있다. Client가 실행중인 Service1의 메모리 사용량은 Service2와 Service3에 비하여 최대 4배정도의 메모리가 사용되고 있으며 Service2의 경우 작업중에는 2400KB~3876KB정도의 메모리를 사용하고 있고 비작업중에는 1700KB정도의 메모리를 요구하고 있다. Service3은 작업중 2000KB정도, 비작업중에는 1000KB미만의 자원만을 요구하고 있다. 이는 작업데이터에 대한 저장과정을 필요로 하지 않고 전송된 데이터에 대한 즉각적인 연산을 수행함으로써 자원의 효율적인 활용을 얻을 수 있다. 따라서 같은 크기의 행렬에 대한 작업을 요구하였을 때 로컬작업에 비하여 많은 작업 소요시간을 요구하지만 자원을 효율적으로 활용할 수 있다.

5. 결 론

본 논문에서는 웹 상에 존재하는 여러 형태의 기기들간의 연합체 구성을 통하여 상호자원을 이용할 수 있는 방안에 대하여 연구하였다.

제안한 웹 기반의 원격 자원 제어 시스템은 네트워크상에 연결된 여러 시스템과 디바이스에 대하여 시간과 공간의 제한없이 접근, 제어할 수 있는 가능성과 함께 구성된 시스템 연합체내의 상호 교류와 접근을 통하여 자원을 제어하고 이용할 수 있는 매카니즘을 제공하여 주었다.

구현된 제어 시스템은 지니 기술을 이용함으로써 적은 비용으로 안정성 있는 시스템 구축방법을 제시하였고 또한 Lookup의 중재 역할로 Client의 서비스 요구에 대한 Service의 상호작업이 원활히 진행될 수 있었다. 실험을 통하여 원격 프로세스의 제어와 원격 자원을 이용한 병렬 프로그래밍의 가능성을 제시하였다.

향후 연구과제로 전체 작업시간을 단축하기 위한 데이터의 전달 방식과 서비스 객체의 등록시 보안에 대한 연구와 함께 다양한 웹 자원활용 방안에 대한 연구가 수행되어야 하리라 사료된다.

참 고 문 헌

[1] A. S. Grimshaw, Wm. A. Wulf, and the Legion team, "The Legion Vision of a WorldWild Virtual Computer," Communications of the ACM, 40(1), January 1997.

[2] K. M. Chandy, B. Dimitrov and H. Le, "A World-Wild Distributed System Using Java and the Internet," In Proceedings of the Fifth IEEE International Symposium on High Performance Distributed Computing, Syacuse, NY Aug. 1996.

[3] Wireless Application Environment Specification, WAP Forum, Nov. 4, 1999, URL : http ://www.wapforulm.org.

[4] Bluetooth URL : http ://www.bluetooth.com.

[5] Universal Plug and Play Forum URL : http ://www.upnp.org.

[6] http ://www.artima.com/jini/booklist.htm.

[7] David Flanagan, "Java in a Nutshell, A Desktop Quick Referecne for Java Programmers," O'Reilly & Associates, Inc, 1996.

[8] S. Oaks, H. Wong, "JINI in a Nutshell," O'Reilly& Associated, Inc March, 2000.

[9] W. Keith Edwards, "Core JINI," Prentice Hall PTR, 1999.

[10] 이정배, 박남섭, "웹을 기반으로 한 원격 UNIX 관리 시스템 구현에 관한 연구", 한국정보처리학회 '98 춘계합동학술논문발표집, 1998.

[11] 이정배, "웹을 기반으로한 원격 제어시스템 환경설계 및 구현", 한국정보처리학회, 1999.

이 기 준

e-mail : cholee@shinbiro.com

1994년 조선대학교 전산통계학과(이학사)

1997년 조선대학교 일반대학원 전산통계학과 (이학석사)

1998년~현재 조선대학교 일반대학원 전산통계학과 박사과정

관심분야 : 신경망, 패턴인식, 인공지능, 분산 에이전트 시스템

정 국 영

e-mail : gyjung@dongkang.ac.kr

1985년 조선대학교 컴퓨터공학과(공학사)

1988년 조선대학교 일반대학원 컴퓨터공학과 (공학석사)

1998년 조선대학교 일반대학원 제어계측공학과 박사과정

1992년 동광대학 컴퓨터정보과 교수

관심분야 : 인공지능, 영상처리, 신호처리

정 채 영

e-mail : cyjung@mail.chosun.ac.kr

1983년 조선대학교 컴퓨터공학과(이학사)

1986년 조선대학교 일반대학원 전자과 전산전공(공학석사)

1989년 조선대학교 일반대학원 전기과 전산전공(공학박사)

1986년~현재 조선대학교 자연과학대학 수학·전산통계학부 부교수

관심분야 : 영상처리, 신경망, 데이터베이스, 멀티미디어 컨텐츠